Forensics Windows : LNK, Prefetch, Amcache et ShimCache analysés

> Guide complet des artefacts forensiques Windows : LNK, Prefetch, Amcache, ShimCache, SRUM, UserAssist — méthodologie d'investigation avec KAPE et Plaso.

Un attaquant peut effacer les journaux d'événements, supprimer ses outils, et nettoyer les traces les plus évidentes de sa compromission. Mais les systèmes Windows enregistrent en permanence, de manière transparente et quasi-impossible à désactiver sans alerter les défenseurs, des dizaines de sources d'artefacts forensiques qui racontent en détail chaque exécution de programme, chaque accès à un fichier, chaque connexion réseau. Les fichiers LNK créés automatiquement quand un utilisateur ouvre un document, les fichiers Prefetch qui tracent l'exécution de chaque programme, l'Amcache qui conserve le SHA1 de tous les exécutables jamais lancés, le ShimCache qui enregistre l'ordre des exécutions depuis le démarrage — chacun de ces artefacts est une pièce du puzzle qui permet de reconstituer fidèlement la chronologie d'une attaque. Pour un analyste SOC, un Incident Responder ou un investigateur forensique, la maîtrise de ces artefacts est la différence entre une investigation qui conclut avec certitude et une qui échoue faute de preuves. Ce guide parcourt méthodiquement les 12 artefacts Windows les plus importants, leurs structures, les informations qu'ils contiennent, les outils pour les analyser, et comment les intégrer dans une timeline d'investigation consolidée.

À RETENIR

À retenir :

Les fichiers Prefetch (.pf) enregistrent les 8 dernières dates d'exécution d'un programme, les fichiers DLL chargés et le nombre d'exécutions — une preuve d'exécution incontestable même si le binaire a été supprimé.

L'Amcache.hve conserve le SHA1 de tous les exécutables ayant jamais tourné sur le système, permettant d'identifier des malwares via VirusTotal même sans avoir le fichier original.

KAPE (Kroll Artifact Parser and Extractor) permet de collecter et parser automatiquement l'ensemble de ces artefacts en quelques minutes sur un système live ou une image forensique.

La corrélation temporelle entre Event Log 4688, Prefetch, Amcache et ShimCache constitue la méthode la plus fiable pour reconstituer une chaîne d'attaque complète.

Pourquoi les artefacts Windows sont des preuves d'investigation irremplaçables

Les systèmes Windows modernes génèrent en permanence des données forensiques issues de dizaines de mécanismes indépendants : gestionnaire de cache applicatif, gestionnaire de compatibilité des applications, service de prefetching, journaux d'événements, ruches de registre utilisateur et système. Ces mécanismes n'ont pas été conçus pour la forensique — ils servent des objectifs fonctionnels précis (améliorer les performances de démarrage, assurer la compatibilité logicielle, faciliter la navigation dans les fichiers récents) — mais leurs enregistrements constituent un journal d'activité système involontaire d'une richesse exceptionnelle.

La valeur forensique de ces artefacts réside dans leur résilience aux tentatives de nettoyage. Un attaquant qui supprime un outil de post-exploitation après usage ne peut pas facilement supprimer l'entrée Prefetch créée lors de son exécution, ni l'entrée Amcache enregistrant son hash SHA1, ni la ligne de l'Event Log 4688 si l'audit de création de processus est activé. Même des artefacts partiellement effacés ou corrompus peuvent livrer des informations exploitables.

Ces artefacts permettent de répondre aux questions forensiques fondamentales : quels programmes ont été exécutés (et quand, combien de fois) ? Quels fichiers ont été accédés ou ouverts ? Des dispositifs externes ont-ils été connectés ? Des connexions réseau sortantes ont-elles eu lieu depuis des processus inhabituels ? Des services ou tâches planifiées malveillants ont-ils été installés ? La combinaison de ces réponses permet de reconstituer une kill chain complète avec des preuves horodatées vérifiables.

Fichiers LNK (Shell Link) : traces des accès aux fichiers

Les fichiers LNK (extension .lnk) sont des raccourcis Windows générés automatiquement par le Shell lorsqu'un utilisateur ouvre un fichier depuis l'Explorateur Windows ou depuis la liste des documents récents. Ils sont stockés dans plusieurs emplacements clés : `%APPDATA%\Microsoft\Windows\Recent\` (fichiers récents), `%APPDATA%\Microsoft\Office\Recent\` (documents Office), et les dossiers épinglés de la barre des tâches.

> ⚡ **Retour terrain**
>
> Dans mes missions forensiques, l'erreur la plus fréquente que je vois est l'isolation du système compromis sans avoir pris d'image mémoire au préalable. Un redémarrage ou une mise hors ligne détruit les artefacts volatils — processus actifs, connexions réseau, clés de chiffrement en mémoire. Ma règle : avant toute action sur un système suspect, dump

mémoire d'abord, isolation ensuite. Cette règle a permis d'identifier l'attaquant dans 3 de mes 5 dernières investigations complexes.

La structure d'un fichier LNK est définie par la spécification Shell Link Binary File Format de Microsoft. Elle contient des informations forensiquement riches : le **chemin absolu de la cible** (y compris les chemins réseau UNC, révélateurs de mouvements latéraux), le **volume serial number** du disque sur lequel le fichier a été accédé (permettant d'identifier si le fichier provenait d'un volume amovible ou d'un partage réseau), les **MAC times** de la cible au moment de la création du LNK (Modified/Accessed/Created — trois timestamps qui peuvent révéler des manipulations temporelles), et la **taille du fichier** cible.

```
 # Analyse de fichiers LNK avec LECmd (Eric Zimmerman)
LECmd.exe -f "C:\Users\Victim\AppData\Roaming\Microsoft\Windows\Recent\malware.lnk"

# Analyse en batch de tous les LNK d'un profil
LECmd.exe -d "C:\Users\Victim\AppData\Roaming\Microsoft\Windows\Recent\" --csv C:\output\ --csvf

# Sous Linux/macOS avec LNKparse3
pip3 install lnkparse3
python3 -c "import lnkparse; l = lnkparse.parse('malware.lnk'); print(l.header.creation_time, l.l
```

Un fichier LNK pointant vers un chemin UNC (`\\10.0.0.50\share\payload.exe`) est une preuve directe d'un accès à une ressource réseau distante — signature classique d'un mouvement latéral. Un LNK pointant vers un lecteur amovible (volume serial number correspondant à une clé USB) peut corroborer une hypothèse d'exfiltration physique. Les **MRU (Most Recently Used)** lists associées aux applications (clés de registre `HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs`) complètent l'information LNK avec le contexte d'accès applicatif.

Prefetch (.pf) : la preuve irréfutable d'exécution de programme

Le service Prefetch Windows (PrefetchParameters dans le registre) enregistre des métadonnées sur les programmes exécutés afin d'accélérer leurs lancements ultérieurs. Chaque fichier Prefetch est stocké dans `C:\Windows\Prefetch\` avec un nom composé du nom de l'exécutable et d'un hash de son chemin complet sur 8 caractères hexadécimaux : `POWERSHELL.EXE-7DE3B8A6.pf`. Ce mécanisme crée des entrées distinctes pour le même binaire exécuté depuis des chemins différents — utile pour détecter des exécutions depuis des répertoires inhabituels (`%TEMP%`, `%APPDATA%`).

Les informations contenues dans un fichier Prefetch sont forensiquement précieuses. Le format Windows 10/11 (version 30) enregistre les **8 dernières dates et heures d'exécution** du programme (pas seulement la dernière), permettant de reconstituer la fréquence d'utilisation d'un outil malveillant sur une période. Le **nombre total d'exécutions** depuis la création de l'entrée est également enregistré. La liste des **fichiers et répertoires référencés** lors de l'exécution inclut les DLLs chargées, les fichiers de configuration accédés, les modules d'extension — révélateurs de l'activité réelle du programme.

```
 # Analyse avec PECmd (Eric Zimmerman)
PECmd.exe -f "C:\Windows\Prefetch\MIMIKATZ.EXE-1B6B1B7E.pf" --mp

# Export CSV de tous les Prefetch
PECmd.exe -d "C:\Windows\Prefetch\" --csv C:\output\ --csvf prefetch_results.csv --mp

# Sur un système live (version portable)
PECmd.exe -d "C:\Windows\Prefetch\" -q --csv . --csvf pf.csv
```

La détection de fichiers Prefetch anormaux est un indicateur de compromission fiable. La présence d'un fichier `MIMIKATZ.EXE-*.pf`, `PROCDUMP.EXE-*.pf`, `PSEXEC.EXE-*.pf` ou `COBALT_STRIKE*.pf` est une preuve directe d'exécution d'outils d'attaque. Même si ces exécutables ont été supprimés après usage, leurs entrées Prefetch persistent et livrent les timestamps d'exécution. La surveillance des nouveaux fichiers Prefetch avec des règles Sigma constitue une méthode de détection efficace à coupler avec les Event Logs — voir notre guide sur les règles Sigma pour la détection.

Amcache.hve : SHA1 des exécutables et détection de binaires inconnus

L'Amcache est une ruche de registre Windows (`C:\Windows\AppCompat\Programs\Amcache.hve`) qui enregistre des métadonnées sur les applications installées et les exécutables qui ont été lancés sur le système. Introduit avec Windows 8, il remplace partiellement l'ancien RecentFileCache.bcf.

La clé forensiquement la plus importante est `Root\InventoryApplicationFile` qui enregistre pour chaque exécutable ayant tourné sur le système : le chemin complet, la taille, le **hash SHA1**, la version, l'éditeur (si signé), et les timestamps de première et dernière exécution. La présence du SHA1 permet une corrélation directe avec des bases de threat intelligence comme VirusTotal ou les feeds IOC internes, même si le fichier original a été supprimé.

```
 # Analyse Amcache avec AmcacheParser (Eric Zimmerman)
AmcacheParser.exe -f "C:\Windows\AppCompat\Programs\Amcache.hve" --csv C:\output\

# Extraction des SHA1 pour lookup VirusTotal en masse
AmcacheParser.exe -f Amcache.hve --csv . -i on
# Les fichiers CSV résultants contiennent les colonnes SHA1 directement interrogeables

# Sous Linux - extraction via regipy
pip3 install regipy
regipy-parse Amcache.hve | grep -i SHA1
```

Un workflow efficace consiste à extraire tous les SHA1 de l'Amcache, les soumettre en batch à l'API VirusTotal (50 requêtes/jour en plan gratuit), et alerter sur tout hash avec un score de détection supérieur à 0. Cette approche permet d'identifier des malwares qui auraient échappé à l'EDR en temps réel — notamment des outils d'attaque légitimes détournés (LOLBins) dont le binaire lui-même n'est pas malveillant mais dont l'usage contextuel l'est.

ShimCache (AppCompatCache) : ordre d'exécution depuis le démarrage

Le ShimCache (officiellement Application Compatibility Cache) est stocké dans la ruche de registre SYSTEM : `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache`. Il enregistre les exécutables qui ont été exécutés (ou simplement "vus" par Windows dans certaines versions) depuis le dernier démarrage du système, dans l'ordre inverse d'exécution — utile pour reconstituer la séquence d'actions d'un attaquant.

Une limitation importante à connaître : **depuis Windows 10, le ShimCache ne garantit plus que les entrées correspondent à des exécutions effectives**. Un fichier peut y apparaître simplement parce que Windows a inspecté son header PE lors d'une opération de copie ou de déplacement. Cette distinction entre "exécuté" (Amcache, Prefetch) et "vu" (ShimCache Windows 10+) est fondamentale pour ne pas tirer de conclusions erronées.

```
 # Extraction avec AppCompatCacheParser (Eric Zimmerman)
AppCompatCacheParser.exe -f "C:\Windows\System32\config\SYSTEM" --csv C:\output\ --csvf shimcache

# Comparaison ShimCache vs Prefetch pour distinguer "vu" vs "exécuté"
# Un fichier dans ShimCache mais absent de Prefetch peut avoir été simplement copié/inspecté
# Un fichier dans Prefetch EST prouvé comme ayant été exécuté
```

La corrélation ShimCache / Prefetch est une technique d'investigation clé : les entrées ShimCache sans entrée Prefetch correspondante peuvent signaler des outils d'attaque préparés mais non encore exécutés (staging de payload), tandis que les entrées Prefetch sans entrée ShimCache (possible si le Prefetch a été nettoyé) peuvent trahir une tentative de couverture de traces.

SRUM : System Resource Usage Monitor et activité réseau

SRUM (System Resource Usage Monitor) est une base de données ESE (Extensible Storage Engine) stockée dans `C:\Windows\System32\sru\SRUDB.dat`. Introduit avec Windows 8, il collecte des données d'utilisation des ressources système toutes les heures et les conserve sur 30 à 60

jours. C'est l'une des sources forensiques les plus riches pour analyser l'activité réseau et applicative sur une période étendue.

Les tables les plus importantes pour la forensique sont : **NetworkUsageMonitor** (bytes envoyés et reçus par application et par interface réseau, avec horodatage), **NetworkConnections** (connexions établies par processus avec adresses IP distantes), et **App_Timeline_Provider** (durée d'utilisation des applications). Ces données permettent de quantifier une exfiltration : si un processus a envoyé 2 GB de données sur une interface réseau le jour de la compromission, c'est une preuve de vol de données quantifiée.

```
 # Analyse SRUM avec SrumECmd (Eric Zimmerman)
SrumECmd.exe -f "C:\Windows\System32\sru\SRUDB.dat" --csv C:\output\

# Extraction des connexions réseau par processus
SrumECmd.exe -f SRUDB.dat --csv . --csvf srum
# Rechercher dans le CSV NetworkConnections les connexions vers des IPs externes inhabituelles
```

UserAssist : compteur d'exécutions via l'interface graphique

UserAssist est une clé de registre dans le profil utilisateur (`NTUSER.DAT`) qui enregistre les applications lancées via l'interface graphique (double-clic, menu Démarrer, barre des tâches). La clé est `HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist` . Les entrées sont encodées en ROT13 — une obfuscation triviale qui n'a jamais été destinée à protéger quoi que ce soit mais qui peut désorienter un analyste débutant.

Chaque entrée contient un compteur du nombre d'exécutions et un timestamp de la dernière exécution. Contrairement au Prefetch (qui trace toutes les exécutions), UserAssist ne trace que les exécutions via l'interface graphique, ce qui en fait un complément utile : un programme lancé en ligne de commande n'apparaîtra pas dans UserAssist mais sera dans Prefetch, permettant de distinguer les actions utilisateur légitimes des exécutions automatisées ou scriptées d'un attaquant.

Jump Lists : historique des fichiers par application

Les Jump Lists sont des listes de fichiers récemment ouverts associées à chaque application Windows, affichées en cliquant-droit sur l'icône d'une application dans la barre des tâches. Elles sont stockées dans deux formats dans le répertoire `%APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations\` et `CustomDestinations\`. Chaque fichier est un format Compound Document identifié par un AppID (identifiant unique de l'application).

Les Jump Lists contiennent des entrées LNK pour chaque fichier récemment ouvert par l'application correspondante, avec les mêmes informations forensiques que les LNK standards (volume serial, MAC times, taille). Elles persistent même après la suppression du fichier original — un document Word supprimé reste référencé dans les Jump Lists de Word jusqu'à ce que la liste soit purgée. L'outil JLECmd d'Eric Zimmerman analyse l'ensemble des Jump Lists et exporte les résultats en CSV.

Windows Event Logs : les journaux clés pour la forensique

Les Event Logs Windows sont stockés dans `C:\Windows\System32\winevt\Logs\` au format EVTX. Les IDs d'événements les plus importants pour l'investigation forensique sont concentrés dans les journaux Security, System et Application.

L'**Event ID 4688** (Process Creation, nécessite l'audit "Audit Process Creation" activé) est le plus précieux : il enregistre la création de chaque processus avec le chemin complet de l'exécutable, les arguments de la ligne de commande (si l'option CommandLine est activée), le PID du processus parent et l'utilisateur ayant lancé le processus. La corrélation de ces logs avec le Prefetch et l'Amcache constitue le socle d'une investigation robuste. Les artefacts de persistance Windows avancés sont analysés dans notre article sur la persistance Windows via registre, COM hijacking et WMI.

Les **Events ID 4624/4625/4648** tracent respectivement les connexions réussies, les échecs d'authentification et les connexions avec des credentials alternatifs (runas) — signatures des mouvements latéraux et des tentatives de pass-the-hash. L'**Event ID 7045** (installation d'un

nouveau service) et l'**Event ID 4698** (création d'une tâche planifiée) sont les indicateurs directs de mécanismes de persistance installés par un attaquant.

```
 # Extraction et analyse avec EvtxECmd (Eric Zimmerman)
EvtxECmd.exe -d "C:\Windows\System32\winevt\Logs\" --csv C:\output\ --csvf evtx_all.csv

# Filtrage des événements critiques avec PowerShell
Get-WinEvent -LogName Security | Where-Object {$_.Id -in @(4688,4624,4648,7045,4698)} |
  Select-Object TimeCreated, Id, Message | Export-Csv events_critical.csv

# Recherche de traces LOLBins dans les Event Logs
# Voir notre guide sur les LOLBins et living off the land
```

La détection de techniques LOLBins (Living Off the Land Binaries) via les Event Logs est documentée dans notre guide sur les LOLBins et living off the land attacks.

Volatility pour l'analyse mémoire complémentaire

L'analyse mémoire avec Volatility complète idéalement l'analyse des artefacts disque. Un dump mémoire (acquired avec WinPmem, DumpIt ou Magnet RAM Capture) contient des informations inaccessibles sur disque : processus en cours d'exécution au moment de l'acquisition (y compris les processus hollowing), connexions réseau établies, clés de chiffrement en mémoire, injections de code dans des processus légitimes.

La corrélation entre artefacts disque et mémoire est particulièrement puissante pour détecter des attaques fileless : un attaquant qui exécute du code PowerShell malveillant directement en mémoire ne laisse pas de trace Prefetch (pas de PE sur disque), mais le dump mémoire révèle le script PowerShell décodé et le processus injecté. Notre article sur l'analyse mémoire forensique avec Volatility détaille ces techniques avancées.

Workflow investigation : timeline consolidée avec KAPE, Plaso et log2timeline

Une investigation forensique Windows efficace repose sur la construction d'une **super-timeline** consolidant tous les artefacts dans un référentiel temporel unique. L'outil **KAPE (Kroll Artifact Parser and Extractor)** est aujourd'hui le standard de facto pour la collecte rapide d'artefacts forensiques Windows. En quelques minutes sur un système live ou une image forensique, KAPE collecte et parse automatiquement les Prefetch, LNK, Jump Lists, Event Logs, Amcache, ShimCache, SRUM et de nombreux autres artefacts.

```
 # Collecte live avec KAPE (en admin)
kape.exe --tsource C: --tdest C:\kape_output --tflush --target !SANS_Triage
# Le target SANS_Triage collecte les artefacts les plus pertinents pour une investigation

# Génération d'une super-timeline avec log2timeline/Plaso
pip3 install plaso
log2timeline.py --storage-file timeline.plaso /path/to/image.E01
psort.py -z UTC -o l2tcsv timeline.plaso > super_timeline.csv

# Analyse avec Timeline Explorer (Eric Zimmerman) ou Excel
# Filtrer par période suspecte, chercher les event_type: 'process_run', 'file_download'
```

La super-timeline résultante, corrélée avec les IoC identifiés (hashes, IPs, noms de fichiers), permet de répondre aux questions forensiques fondamentales avec des preuves horodatées : à quelle heure précise l'outil malveillant a-t-il été exécuté pour la première fois ? Quels fichiers a-t-il accédés dans les minutes suivantes ? Des connexions réseau sortantes vers des IPs C2 peuvent-elles être mises en corrélation temporellement avec l'exécution du malware ? Cette rigueur méthodologique est la marque d'une investigation forensique professionnelle.

FAQ — Questions fréquentes

Comment savoir si un attaquant a tenté de supprimer les fichiers Prefetch ?

La suppression manuelle des fichiers Prefetch est elle-même une indicator of compromise. Premièrement, l'absence de fichiers Prefetch sur un système Windows 10/11 opérationnel est anormale — Windows en génère en permanence lors des utilisations normales. Deuxièmement, le journal des événements Windows (Event ID 4663 si l'audit des objets est activé sur `C:\Windows\Prefetch\`) peut enregistrer les suppressions de fichiers. Troisièmement, des outils comme Sysmon (Event ID 23 - FileDelete) tracent les suppressions de fichiers avec le hash de chaque fichier supprimé, permettant de récupérer le nom et le hash du fichier Prefetch supprimé même sans le fichier lui-même. Quatrièmement, les entrées ShimCache et Amcache peuvent encore prouver qu'un exécutable a tourné même si son Prefetch a été supprimé.

Les artefacts forensiques sont-ils fiables dans un environnement Windows virtualisé ou VDI ?

Les environnements VDI (Virtual Desktop Infrastructure) présentent des spécificités forensiques importantes. Dans les architectures de bureaux non-persistants (stateless VDI, sessions Citrix XenApp), les artefacts forensiques sont effacés à chaque déconnexion — ce qui complique considérablement les investigations. En revanche, les VDI persistants se comportent comme des systèmes physiques standard. Pour les environnements cloud (Azure Virtual Desktop, AWS WorkSpaces), les logs de la plateforme cloud (Azure Monitor, AWS CloudTrail) compensent partiellement l'absence d'artefacts locaux. La présence d'un EDR comme Wazuh ou CrowdStrike avec télémétrie cloud est essentielle dans ces environnements car il constitue la seule source de preuves d'exécution fiable sur les VDI non-persistants.

Quels outils utiliser pour une collecte forensique Windows en urgence (first responder) ?

Pour un first responder intervenant sur un système potentiellement compromis, le kit minimal recommandé comprend : **KAPE** sur une clé USB avec le target SANS_Triage (collecte automatisée en 5-10 minutes), **WinPmem** pour l'acquisition mémoire avant toute autre action (la mémoire est volatile et sa valeur diminue avec chaque minute d'utilisation du système), et

IRTriage-Master ou **CyLR** pour la collecte complémentaire. L'ordre des opérations est critique : mémoire en premier (la plus volatile), puis artefacts disque (Prefetch, LNK, Event Logs), puis image disque complète si nécessaire. Éviter absolument d'installer des outils d'analyse sur le système compromis — tout doit s'exécuter depuis la clé USB pour ne pas écraser des artefacts ou modifier les timestamps d'accès aux fichiers.

Sources et références

MITRE ATT&CK — Techniques de collecte et d'investigation

ANSSI — Guide d'investigation numérique
